Random Numbers

Author: Elise Baldwin

Assignment Definition And General Feedback By Michael Panitz
at Cascadia Community College (http://www.cascadia.edu)

Table of contents:

Quick Summary:

In Java, you can create random number generators using the java.util.Random class. Though they aren't truly random, a generator can be made to create a sequence of unpredictable numbers within a given range. You can then create one of many types of variables to be assigned to variables, includinginteger, Boolean, double and Gaussian variables. This document will be focusingexclusively on integers.

When To Use This / Avoid This:

This technique can be used any time you want to assign a random value to an integer. Some examples in which this would be useful are:

- To make a robot move in a random direction.
- To make a robot move a random number of steps forward.
- To randomly place walls for the robot to avoid.
- To place a robot in an unknown location and send another robot after it.
- Send a virtual robot named Asimov after a randomly placed lemon cake.
- To randomly print text in the output window in order to confuse the user for no apparent reason.

The second-to-last example will be demonstrated in the following section. Note that in the case of the last example, the code provided here would be assigned to pre-written lines of code. The random number generators discussed here only use integers— it cannot be used for decimals or text.

Example Of Usage:

As an example, say you want to create a thing placed randomly in the city, and a robot which can find and pick it up. The first task is to generate random street and avenue values and create a thing there. First, in the top line of code (in the same place as`import becker.robots.*;`), enter this line:

```
import java.util.Random;
```

This provides access to the files necessary to create random numbers. Second, create a random number generator:

```
Random generator = new Random();
```

Like any other object, it can have any legal name, including “generator”. The parenthesis can be empty, as shown, but it can also include a seed. This seed is the number from which the generator starts its sequence of numbers, which are actually random but are based on a complicated algorithm, called a linear congruential generator, that make them appear to be. Note that if the seed is given here, the generator will display the same number every time, so it may be better in most cases to leave it out.

The generator itself is not a number, but can be used to create one. This is the third and final step, which can be done by typing:

```
generator.nextInt();
```

It is a good idea to assign this to a specific variable, such as:

```
int num = generator.nextInt();
```

Note that by itself, this line of code can create very large numbers (both positive and negative). It is helpful to set the upper limit for the numbers. For example, to select a random number between 0 and 9, enter:

```
int num = generator.nextInt(10);
```

In this case, num can be one integer below ten (nine), zero, or any integer in between. This limiting value can be any integer, as long as it is greater than zero.

Note that you can change the range by multiplying the variable by numbers or adding numbers to it. Adding a number to it moves the range up (for example, adding two would move the range form 0-9 to 2-11). Multiplying increases the range (for example, multiplying by three changes the range from 0-9 to 0-27). To get a random integer in the range 5-12, use:

```
intnum = generator.nextInt(8) + 5;
```

Try practicing this for different ranges, using the print command to see what results you come up with.

Now this will be applied to the program itself. In lines 14 and 15, the generator will assign random numbers to the variables randStreet and randAvenue, and on line 16 they will be entered into the line which creates a thing. The ranges for these will be set so the thing appears reasonable close to the robot (five, rather than 3141592 intersections away). The robot will then be programmed to move until it is in the same location, and pick up the thing.

Line	Code
1.	`import becker.robots.*;`
2.	`import java.util.Random;`
3.	
4.	`public class Random_Thing extends Object`
5.	`{`
6.	`  public static void main(String[] args)`
7.	`{`
8.	`//Create city and robot`
9.	`  City flatland = new City(11,11); //Display 11 streets and avenues (0-10)`
10.	`  Robot asimov = new Robot(flatland, 0, 0, Direction.EAST, 0);`
11.	
12.	`  //Generate random street and avenue, create unpredictable lemon cake`
13.	`  Random generator = new Random();`
14.	`int randStreet = generator.nextInt(10) + 1; //Streets between 1 and 10`
15.	`int randAvenue = generator.nextInt(10) + 1; //Same for avenues`
16.	`  Thing lemon_cake = new Thing(flatland, randStreet, randAvenue);`
17.	
18.	`  //Find lemon cake`

19.	`while(asimov.getAvenue() <randAvenue)`
20.	`{`
21.	`asimov.move();`
22.	`}`
23.	`asimov.turnLeft();`
24.	`asimov.turnLeft();`
25.	`asimov.turnLeft();`
26.	`while(asimov.getStreet() <randStreet)`
27.	`{`
28.	`asimov.move();`
29.	`}`
30.	`asimov.pickThing();`
31.	`}`
32.	`}`

Example Trace:

In a nutshell, this technique allows you to make as many random integers as needed for a program. In order to go over these details more thoroughly, here is a partial trace of the above program, including notes for some of the steps. For this example, the generator will give the coordinates (4, 2) for the thing, though this is just one of a hundred possibilities.

Line #	Program Statement	asimov's Location (St, Ave)	asimov's Direction	randStreet	randAvenue	Thing's Location (St, Ave)	Notes
2	`importjava.util.Random;`	-	-	-	-	-	Provides access to files necessary for random number generators on line 13.
9	`City flatland = new City(11,11);`	-	-	-	-	-	Create city.
10	`Robot asimov = new Robot(flatland, 0, 0, Direction.EAST, 0);`	(0,0)	E	-	-	-	Create robot.
13	`Random generator = new Random();`	(0,0)	E	-	-	-	Create generator. Note that it is not an actual number value yet.
14	`intrandStreet =`	(0,0)	E	4	-	-	Generator gives

	generator.nextInt(10) + 1;						randStreet a value.
15	intrandAvenue = generator.nextInt(10) + 1;	(0, 0)	E	4	2	-	Generator gives randAvenue a value.
16	Thing lemon_cake = new Thing(flatland, randStreet, randAvenue);	(0, 0)	E	4	2	(4, 2)	randStreet and randAvenue used to create thing (devious lemon cake).
19	while(asimov.getAvenue() <randAvenue)	(0, 0)	E	4	2	(4,2)	Asks if robot is to the left of the thing. It is, so it continues.
21	asimov.move();	(0, 1)	E	4	2	(4, 2)	Robot moves closer to thing.
19	while(asimov.getAvenue() <randAvenue)	(0, 1)	E	4	2	(4, 2)	Repeat loop.
21	asimov.move();	(0, 2)	E	4	2	(4, 2)	
19	while(asimov.getAvenue() <randAvenue)	(0, 1)	E	4	2	(4, 2)	Loop complete. Robot is directly north of thing.
23	asimov.turnLeft();	(0, 2)	N	4	2	(4, 2)	Robot turns to face south.
24	asimov.turnLeft();	(0, 2)	W	4	2	(4, 2)	
25	asimov.turnLeft();	(0, 2)	S	4	2	(4, 2)	
26	while(asimov.getStreet() <randStreet)	(0, 2)	S	4	2	(4, 2)	Tests if robot is still north of thing. It is.
28	asimov.move();	(1, 2)	S	4	2	(4, 2)	Robot moves again.
26	while(asimov.getStreet() <randStreet)	(1, 2)	S	4	2	(4, 2)	The loop goes on.
28	asimov.move();	(2, 2)	S	4	2	(4, 2)	
26	while(asimov.getStreet() <randStreet)	(2, 2)	S	4	2	(4, 2)	And goes on.
28	asimov.move();	(3, 2)	S	4	2	(4, 2)	
26	while(asimov.getStreet() <randStreet)	(3, 2)	S	4	2	(4, 2)	
28	asimov.move();	(4, 2)	S	4	2	(4, 2)	
26	while(asimov.getStreet() <randStreet)	(4, 2)	S	4	2	(4, 2)	Finally, the loop is complete. The robot is with the thing.
30	asimov.pickThing();	(4, 2)	S	4	2	-	Mission accomplished.

Syntax Explanation:

Overall, the three steps which are absolutely necessary to create a random integer are:

- **Starting the program with the line:**

```
importjava.util.Random;
```

- **Creating a generator:**

```
Random <generator name> = new Random();
```

- **Assigning a value to the desired variable:**

```
int<variable name> = <generator name>.nextInt();
```

The final variable can then be used in the same way as a normal integer would be used. The same rules apply to these lines of code that apply to all others: the generator and variable names must be legal (they must be a series of letters and numbers with no spaces, and can't begin with a number); the opening and closing parenthesis must be correctly placed; there must be a semicolon at the end of each line.

Hyperlinks

Research for this document was greatly facilitated by the following two internet resources. If for some reason this explanation did not make sense, or you would like to find out more about the topic, please take a look at one or both websites.

"Random Numbers in Java" by Doug Baldwin (no relation):
http://www.cs.geneseo.edu/~baldwin/reference/random.html

Oracle's definition "Random" class:
http://download.oracle.com/javase/1.4.2/docs/api/java/util/Random.html

Licensing

Plagiarism

If you believe that some or all of this document infringes on your intellectual property (i.e., part or all of this document is copied from something you've written) please immediately contact Mike Panitz at Cascadia Community College (perhaps using the Faculty And Staff Directory at http://www.cascadia.edu/pages/searchtemplate.aspx)